Imagination

THE IMAGINATION UNIVERSITY PROGRAMME

RVfpga-SoC
Installation Guide

Table of Contents

1. Introduction

This guide shows how to install tools and hardware needed for RVfpga-SoC on the Ubuntu 18.04 operating system (OS). **The instructions below are for an Ubuntu 18.04 OS**, but other Linux operating systems and Windows follow similar (if not exactly the same) steps. We insert boxes with specific instructions for the Windows OS in some cases. If you are using Ubuntu, ignore those boxes.

This process can take several hours (or more, depending on your download speed), but most of the time is spent waiting while the programs are downloaded and installed.

Table 1 lists the software and hardware needed for RVfpga-SoC.

Table 1. Required Software and Hardware for RVfpga-SoC

Software		
Name	**Website**	**Cost**
Vivado 2019.2 WebPACK	https://www.xilinx.com/support/download/index.html/content/xilinx/en/downloadNav/vivado-design-tools/2019-2.html	free
VSCode	https://code.visualstudio.com/Download	free
PlatformIO	https://platformio.org/ Installed within VSCode	free
Verilator (an HDL simulator) and GTKWave	https://github.com/verilator/verilator http://gtkwave.sourceforge.net/	free
FuseSoC	https://github.com/olofk/fusesoc	free
RISC-V Toolchain and OpenOCD	https://github.com/riscv/riscv-gnu-toolchain https://github.com/riscv/riscv-openocd Installed within PlatformIO	free
Zephyr Project	https://github.com/zephyrproject-rtos/zephyr	free
Hardware*		
Name	**Website**	**Cost**
Nexys A7 FPGA Board*	https://store.digilentinc.com/nexys-a7-fpga-trainer-board-recommended-for-ece-curriculum/	$341 (academic price: $261.75)
RISC-V Core and System-on-Chip (SoC)**		
Name	**Website**	**Cost**
Western Digital's SweRV EH1 Core	https://github.com/chipsalliance/Cores-SweRV	free
SweRVolf	https://github.com/chipsalliance/Cores-SweRVolf	free

* Hardware is optional.
** The SweRV EH1 core and SweRVolf are provided as part of the RVfpga-SoC package.

2. Installation for Lab 1

This section will show you how to install the software required for performing Lab 1 of the RVfpga-SoC course.

1. Install Vivado:

Vivado is a Xilinx tool for viewing, modifying, and synthesizing Verilog code. You will use it extensively in later labs. The installation instructions are available at
https://reference.digilentinc.com/vivado/installing-vivado/start and are summarized below.

> **Windows:** the webpage referenced above
> (https://reference.digilentinc.com/vivado/installing-vivado/start) also includes detailed instructions for installing Vivado in Windows. Below we insert boxes when specific instructions are required for Windows.

Step 1. Navigate to https://reference.digilentinc.com/vivado/installing-vivado/start

Step 2. You will be guided to the Xilinx download page:
https://www.xilinx.com/support/download.html

Step 3. It is recommended that you install the "Self Extracting Web Installer". At the time of writing this document, it is at this link on the download page: Xilinx Unified Installer 2019.2: Linux Self Extracting Web Installer.

> **WINDOWS:** At the time of writing this document, the "Self Extracting Web Installer" for Windows is at this link on the download page: Xilinx Unified Installer 2019.2: Windows Self Extracting Web Installer

Step 4. You will be asked to log in to your Xilinx account before you can download the installer. If you don't already have an account, you will need to create one.

Step 5. Execute the binary file. Open a terminal and make it root (type "sudo su"). Then drag the binary file (Xilinx_Unified_2019.2_1106_2127_Lin64.bin) into the terminal. If it prompts you to make the file executable and run it, select OK.

Troubleshooting: If the terminal says permission denied, type the following in the terminal (in the same directory as the binary file):
> ➢ `sudo chmod +x ./Xilinx_Unified_2019.2_1106_2127_Lin64.bin`
> ➢ `sudo ./Xilinx_Unified_2019.2_1106_2127_Lin64.bin`

> **WINDOWS:** In Windows, you can simply execute the .exe file that you downloaded in steps 3 and 4 by double-clicking on it.

Step 6. The Vivado installer will walk you through the installation process. Important notes:
- Select **Vivado** (*not* Vitis) as the Product to install.
- Select Vivado HL **Webpack** (*not* Vivado HL System Edition); Webpack is free.
- Otherwise, defaults should be selected.

> **Hint:** If you change the installation directory of Vivado, you would need to modify the path appropriately in the following steps.

WINDOWS: Steps 7 and 8 are not necessary for Windows. You
can simply ignore these two steps and go directly to step 9.

Step 7. After Vivado has been installed, you need to set up the environment. Open a
terminal and type:

> ➢ `source /tools/Xilinx/Vivado/2019.2/settings64.sh`

   Add that line (`source /tools/Xilinx/Vivado/2019.2/settings64.sh`) to
your `~/.bashrc` file so that it runs each time you launch a terminal.

Step 8. Test Vivado by typing the following in a terminal:

> ➢ `vivado`

Troubleshooting:
- If your system cannot find that executable, you'll need to add the following to your
  path:

  ```
  /tools/Xilinx/DocNav
  /tools/Xilinx/Vivado/2019.2/bin
  ```

- If you get an error such as "application-specific initalization failed…", type the
  following at a terminal:

  > ➢ `sudo ln -s /lib/x86_64-linux-gnu/libtinfo.so.6`
  > `/lib/x86_64-linux-gnu/libtinfo.so.5`

2. Install Cable drivers:
Step 9. You will need to **manually install the Nexys A7 FPGA board's cable drivers**.
Type the following at a terminal window:
> ➢ `cd`
> `/tools/Xilinx/Vivado/2019.2/data/xicom/cable_drivers/`
> `lin64/install_script/install_drivers/`
>
> ➢ `sudo ./install_drivers`

WINDOWS: Vivado installation in Windows automatically installs drivers for the Nexys
A7 board, which are not compatible with PlatformIO. Thus, if you are using Windows,
you must update the drivers as explained in Appendix A of the Installation Guide.

3. Install Digilent Board Files:
You will also need to install the Digilent Board Files manually.
Step 10. Download the [archive](#) of the vivado-boards from the Github repository and
extract it.

Step 11. Open the folder extracted from the archive and navigate to its *new/board_files*
directory. Select all folders within this directory and copy them.

Step 12. Open the folder that Vivado was installed into (*/tools/Xilinx/Vivado* by default).
Under this folder, navigate to its *<version>/data/boards/board_files* directory, then paste
the board files into this directory.
Step 13. You can also use the terminal, by going into the *new/board_files* directory and
typing:

➢ `sudo cp -r * `
 `/tools/Xilinx/Vivado/2019.2/data/boards/board_files`

> **WINDOWS:** copy/paste the downloaded folders as explained in Step 10. In Windows, you can find Vivado's *board_files* folder at:
> *C:\Xilinx\Vivado\2019.2\data\boards\board_files*

3. Installation for Lab 2

This section will show you how to install the software required for performing Lab 2 of the RVfpga-SoC course. It includes instructions for installing software such as Visual Studio Code (VSCode), PlatformIO, Verilator, and GTKWave.

1. Install VSCode:
Follow these steps to install VSCode:
Step 1. Download the .deb file from the following link:
https://code.visualstudio.com/Download

Step 2. Open a terminal, and install and execute VSCode by typing the following in the terminal:
➢ `cd ~/Downloads`
➢ `sudo dpkg -i `<filename>`.deb` (Change the filename)
➢ `code`

> **Windows:** VSCode packages are also available for Windows (.exe file) at
> https://code.visualstudio.com/Download. Follow the common steps used for installing and executing an application in these operating systems.

2. Install PlatformIO on top of VSCode:
PlatformIO is an integrated development environment (IDE) for embedded systems that is built on top of Microsoft's Visual Studio (VS) Code. It allows you to program the RISC-V processor (that is located on the FPGA) using C or assembly. PlatformIO is cross-platform and includes a built-in debugger.

Follow these steps to install PlatformIO:
Step 3. Install python3 utilities by typing the following in a terminal:

➢ `sudo apt install -y python3-distutils python3-venv`

> **Windows:** this step 3 is not required in Windows.

Step 4. If not yet open, start VSCode by selecting the Start button and typing "VSCode" in the search menu, then select VSCode, or by typing `code` in a terminal.

Step 5. In VSCode, click on the Extensions icon   located on the left sidebar of VSCode (see Figure 1).

Figure 1. VSCode's Extensions icon

Step 6. Type *PlatformIO* in the search box and install the PlatformIO *IDE* by clicking on the install button next to it (see Figure 2).
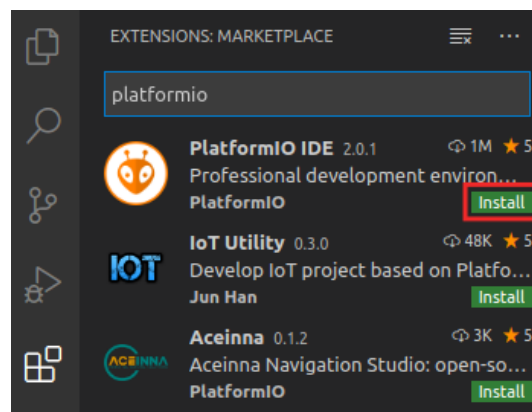

Figure 2. PlatformIO IDE Extension

Step 7. The OUTPUT window on the bottom will inform you about the installation process. Once finished, click "Reload Now" on the bottom right side window, and PlatformIO will be installed inside VSCode (see Figure 3).
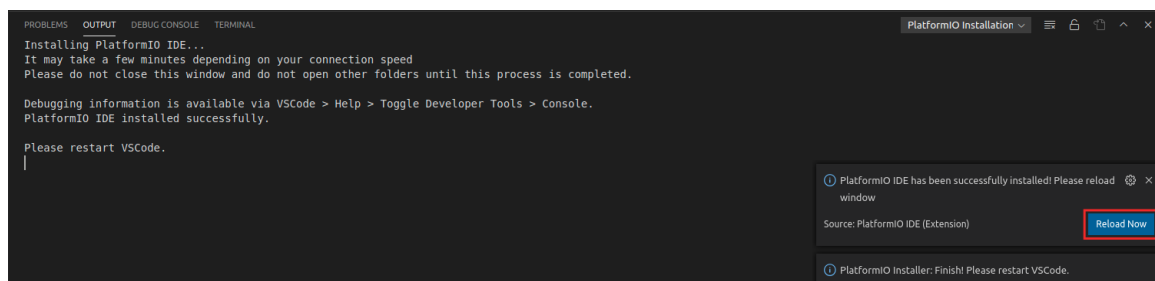

Figure 3. Reload Now after PlatformIO installs

The first time that an example is opened in PlatformIO, the Chips Alliance platform gets automatically installed (you can view it inside the PIO Home, as shown in Figure 4). This platform includes several tools that you will use later, such as the pre-built RISC-V toolchain, OpenOCD for RISC-V, an RVfpgaNexys bitfile and RVfpgaSim, JavaScript and Python scripts, and several examples.
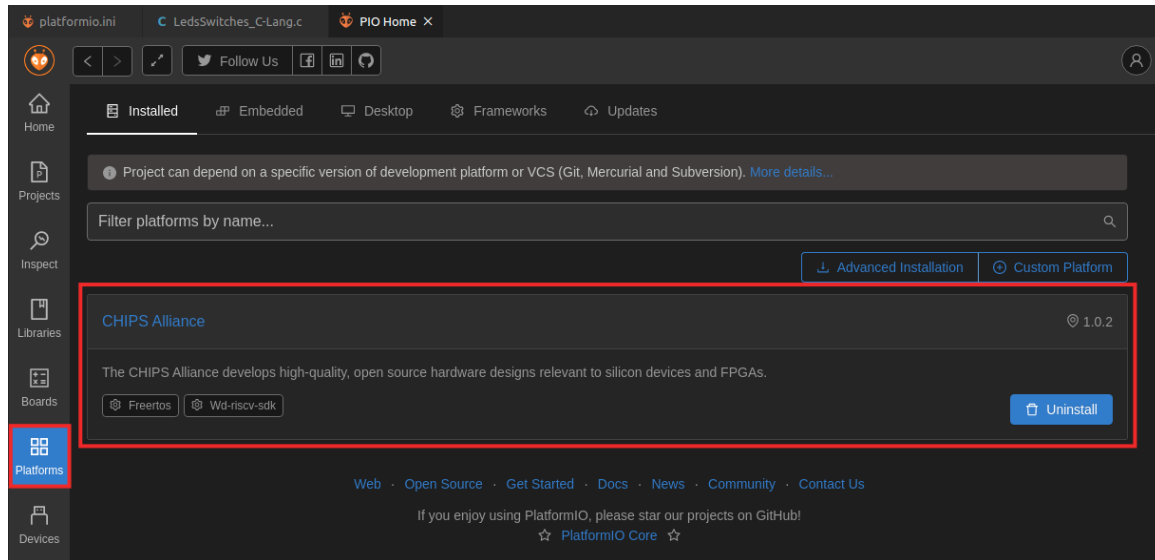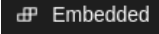
Figure 4. Chips Alliance platform installed in PlatformIO

If for any reason, the Chips Alliance platform did not install automatically when you open an example in Lab 2, you can install it manually following the next steps:

- View the Quick Access menu by clicking on the [icon] button, located in the left side bar (see Figure 5). Then, in the PIO Home, click on the [Platforms] button and then on the [Embedded] tab (Figure 5). Look for **Chipsalliance** (the platform that we use in RVfpga) and open it by clicking on the [CHIPS Alliance] button (Figure 5).
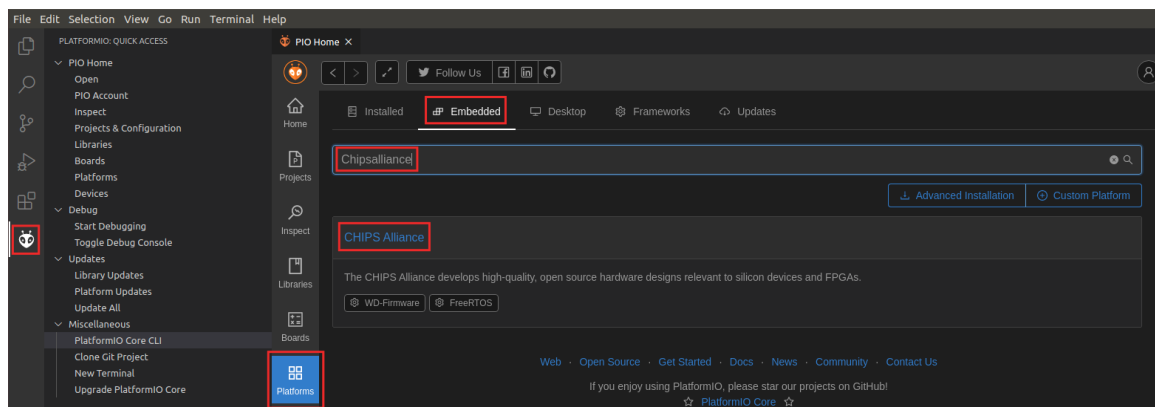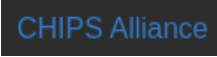
Figure 5. Selecting the CHIPS Alliance Platform

- After clicking on the [CHIPS Alliance] button, you will see the details of the Chips Alliance platform (as in Figure 6). Install it by clicking on the [Install] button (Figure 6).
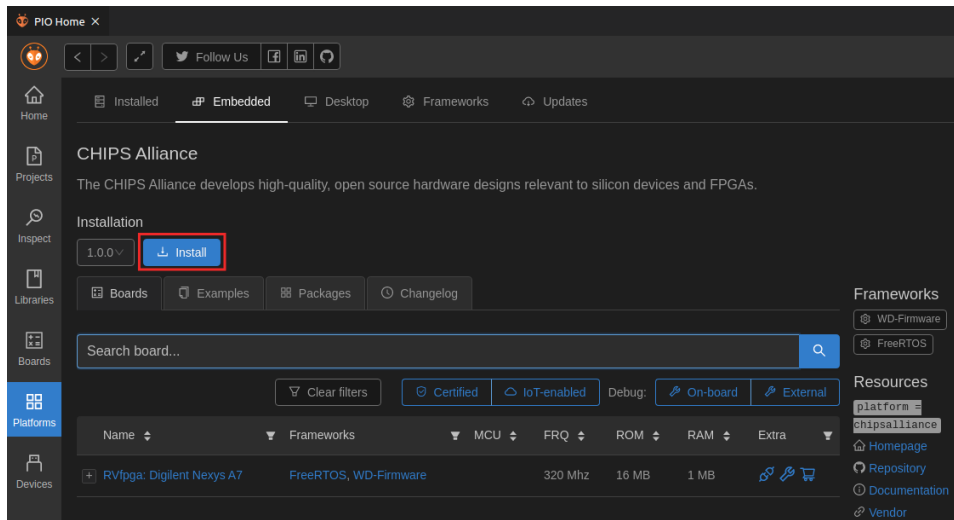
Figure 6. Installing the CHIPS Alliance Platform

- Once installation completes, a summary of the tools that have been installed is shown, as in Figure 7. Click OK to close that window.
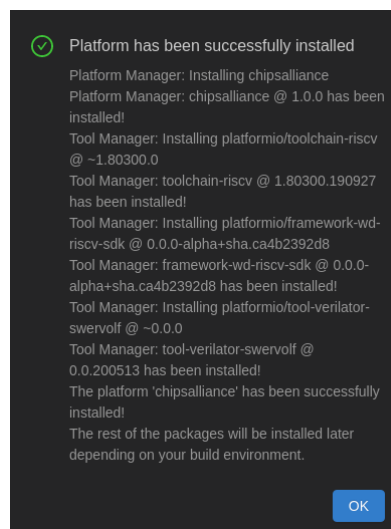

Figure 7. Successful installation of CHIPS Alliance Platform

3. Install GTKWave:
Follow the next steps to install GTKWave in your Ubuntu 18.04 Linux system.
Open your Ubuntu terminal and enter the following commands :

```
□ sudo apt-get install git make autoconf g++ flex bison
  libfl2 libfl-dev
□ sudo apt-get install -y gtkwave
```

> **Windows:** See Appendix B for the Installation of GTKWave for Windows.

4. Install Verilator:
Follow the next steps to install Verilator (instructions are available at https://www.veripool.org/projects/verilator/wiki/Installing but are also summarized below)

```
☐  git clone https://github.com/verilator/verilator
☐  cd verilator
☐  git pull
☐  git checkout v4.106
☐  autoconf
☐  ./configure
☐  make    (alternatively, you can use make -j$(nproc) to make it go faster)
☐  sudo make install
☐  export PATH=$PATH:/usr/local/bin (change the path in your system)
```

To add *usr/local/bin* permanently to your path, add the last line to your ~/.bashrc file.

Note: It may take a while to install depending on your computer.

Windows: See Appendix B for the Installation of Verilator for Windows.

4. Installation for Lab 3

The Installation instructions from Lab 3 onwards are specifically for Ubuntu 18.04 operating system. Windows 10 users can run the simulation parts of the labs using Windows Subsystem for Linux. The instructions will work the same for any other latest version of Ubuntu. It is highly recommended that you perform all these following labs on the Ubuntu operating system.

This section will show you how to install the software required for performing Lab 3 of the RVfpga-SoC course.

1. Install pip:
"**pip**" is required for "FuseSoC" and "west" Installation. Open the Ubuntu terminal and enter the following command :

➢ `sudo apt install python3-pip`

2. Install pyelftools:
pyelftools is required for west build. We can Install pyelftools using 2 ways:

➢ pip3 install pyelftools
    or
➢ sudo apt-get install -y python3-pyelftools python-pyelftools

3. Install FuseSoC:
To install the current stable version of FuseSoC, open a terminal window and run the following command. If an older version of FuseSoC is found on your system, this will upgrade the version to the latest stable release.

➢ `sudo pip3 install --upgrade fusesoc`

Note: Performing "apt upgrade" has 261 packages. Expect longer processing time. ⏳

4. Install OpenOCD:

OpenOCD is an open on-chip debugger that allows users to
program and debug embedded target devices. Follow the next steps to install RISC-V
OpenOCD onto your computer:

Step 1. Use "apt-get" to install the required dependencies:
- ➢ `sudo apt-get install libusb-1.*`
- ➢ `sudo apt-get install pkg-config`
- ➢ `sudo apt-get install libtool`

Step 2. Clone the riscv-openocd github repository:

- ➢ `git clone https://github.com/riscv/riscv-openocd.git`
- ➢ `cd riscv-openocd`
- ➢ `./bootstrap`

Step 3. Download and install the userspace USB programming library development files.

- ➢ `sudo apt-get install libusb-1.0-0-dev`

Step 4. Configure the JTAG server which the OpenOCD can connect to
- ➢ `./configure --enable-jtag_vpi --enable-ftdi`
- ➢ `make`
- ➢ `sudo make install`

5. Installation for Lab 4

This section will show you how to install the software required for performing Lab 4 of the
RVfpga-SoC course.

Open your Ubuntu terminal and enter the following commands :

1. Prerequisites and Dependencies:
Step 1. Update the repositories by using :
- ➢ `sudo apt update`
- ➢ `sudo apt upgrade`

Note: Executing "apt upgrade" on Ubuntu 18.04 upgrades 261+ packages. Expect longer
processing time.

Step 2. Use "apt" to install the required dependencies:
- ➢ `sudo apt install --no-install-recommends git cmake ninja-build gperf \`
  - ○ `ccache dfu-util device-tree-compiler wget \`
  - ○ `python3-dev python3-pip python3-setuptools python3-tk python3-wheel xz-utils file \`
  - ○ `make gcc gcc-multilib g++-multilib libsdl2-dev libmagic1`

Figure 8. Dependencies

2. Update *cmake* version to 3.13.1 or higher:
You can follow the instructions for adding the [kitware third-party apt repository](#) to get an updated version of cmake using apt.

Step 3. Enter the following commands to download and install/update cmake:

➢ `wget -O - https://apt.kitware.com/keys/kitware-archive-latest.asc 2>/dev/null |`
➢ `sudo apt-key add -`
➢ `sudo apt-add-repository 'deb https://apt.kitware.com/ubuntu/ bionic main'`
➢ `sudo apt update`
➢ `sudo apt install cmake`

Note: If you are unable to update "cmake" using the instructions above? No worries! Visit the link below for alternative instructions on how to update "cmake" and ensure you have the latest version. Simply follow the steps provided for a seamless update process.

Link:
https://askubuntu.com/questions/355565/how-do-i-install-the-latest-version-of-cmake-from-the-command-line

3. Install west:
Step 4. Install west, and make sure ~/.local/bin is on your PATH environment variable.

➢ `pip3 install --user -U west`
➢ `echo 'export PATH=~/.local/bin:"$PATH"' >> ~/.bashrc`
➢ `source ~/.bashrc`

4. Install Zephyr SDK :
The Zephyr Software Development Kit (SDK) contains toolchains for each of Zephyr's supported architectures. It also includes additional host tools, such as custom QEMU binaries and a host compiler. We will Install the Zephyr SDK version 0.12.4

Enter the following commands in your **home** directory:

Step 5. Download the <u>0.12.4 version SDK installer</u>:

> ➢ `wget`
>   `https://github.com/zephyrproject-rtos/sdk-ng/releases/down`
>   `load/v0.12.4/zephyr-sdk-0.12.4-x86_64-linux-setup.run`

Step 6. Run the installer, installing the SDK in `~/zephyr-sdk-0.12.4`:

> ➢ `chmod +x zephyr-sdk-0.12.4-x86_64-linux-setup.run`

> ➢ `./zephyr-sdk-0.12.4-x86_64-linux-setup.run -- -d`
>   `~/zephyr-sdk-0.12.4`

If installing the Zephyr SDK outside any of those locations, please read: Install the Zephyr Software Development Kit (SDK). You cannot move the SDK directory after you have installed it.

5. Install PuTTY :
Step 7. Enter the following command to install PuTTY :

> ➢ `sudo apt-get install -y putty`

6. Appendix A: Installing drivers in Windows to use PlatformIO

To download the Zadig executable, browse to the following website (see Figure 9):

  https://zadig.akeo.ie/
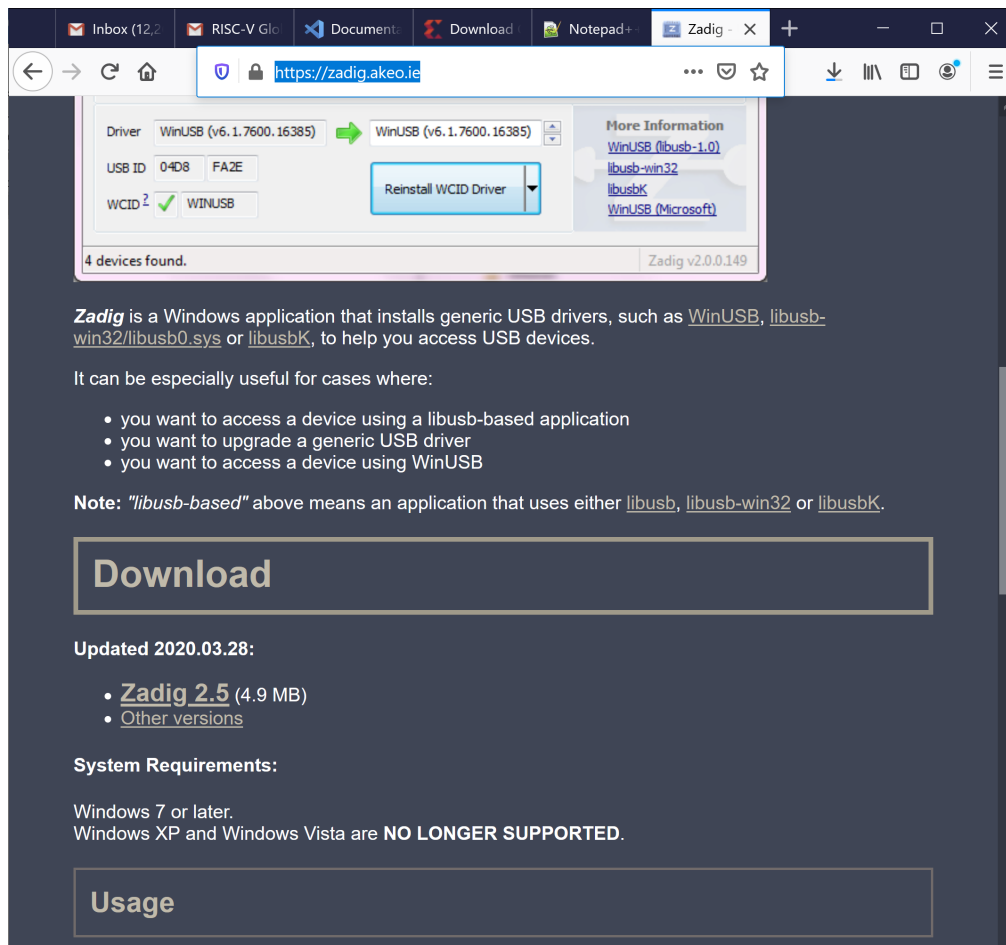
Figure 9. Install Nexys A7 board driver used by PlatformIO

Click on Zadig 2.5 and save the executable. Then run it (zadig-2.5.exe), which is located where you downloaded it. You can also type zadig into the Start menu to find it. You will probably be asked if you want to allow Zadig to make changes to your computer and if you will let it check for updates. Click Yes both times.

Connect the Nexys A7 Board to your computer and switch it on. In Zadig, click on Options → List All Devices (see Figure 10).
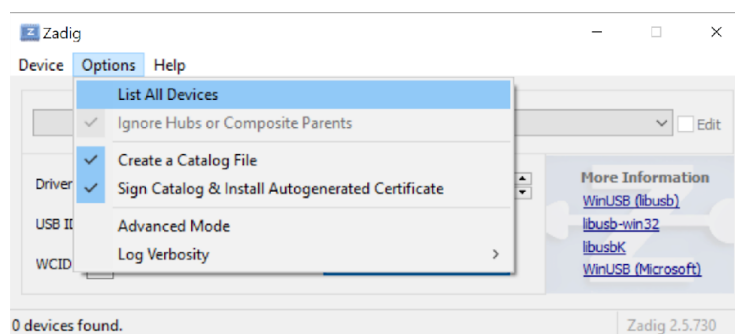
Figure 10. List all devices in Zadig

If you click on the drop-down menu, you will see Digilent USB Device (Interface 0) and Digilent USB Device (Interface 1) listed. You will install new drivers for only Digilent USB Device (Interface 0) (see Figure 11).
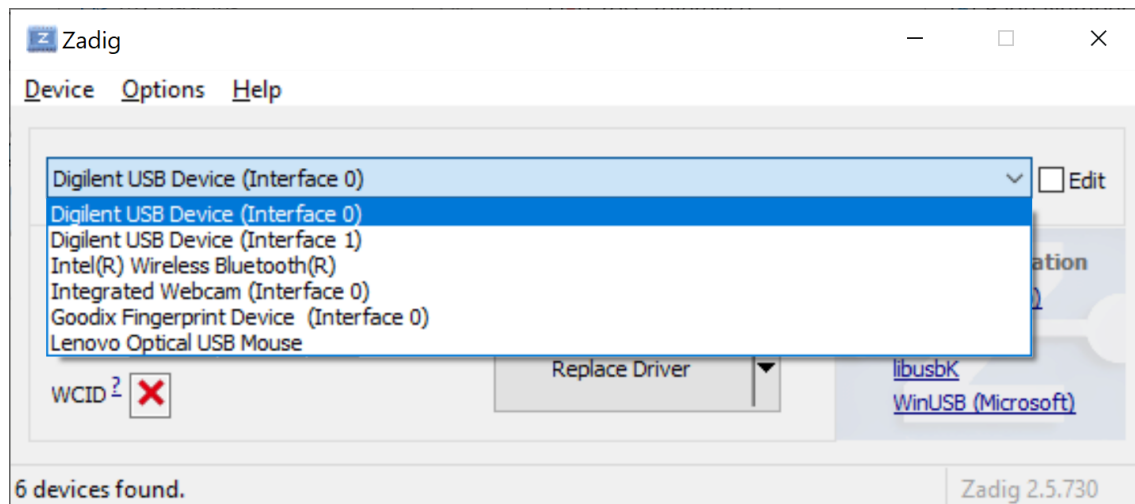
Figure 11. Install WinUSB driver for Digilent USB Device (Interface 0)

You will now replace the FTDI driver with the WinUSB driver, as shown in Figure 12. Click on Replace Driver (or Install Driver) for Digilent USB Device (Interface 0). You are installing the driver for the Nexys A7 board or, if you previously installed Vivado, you are replacing the FTDI driver used by Vivado with the WinUSB driver used by PlatformIO.
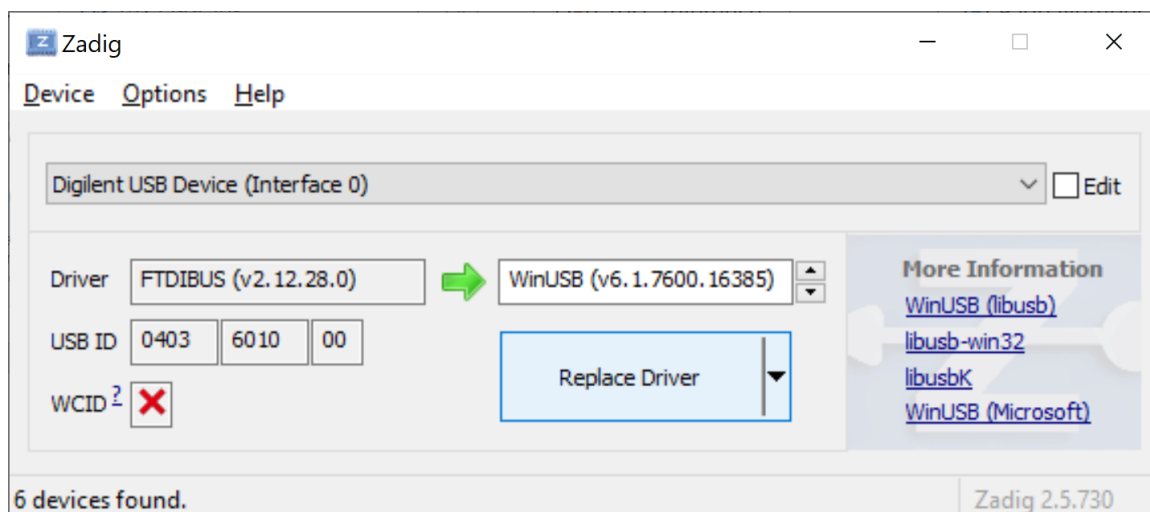

Figure 12. Replace driver for Nexys A7 board

After some time, typically several minutes, Zadig will indicate the driver was installed correctly. Click Close and then close the Zadig window.

Next time you use PlatformIO you do not need to re-install the driver. However, note that **this driver is not compatible with Vivado in Windows**.

7. Appendix B: Installing Verilator and GTKWave in Windows

In this section, we explain how to install Verilator and GTKWave in Windows 10. In Windows, you must use Cygwin to install Verilator, so we first explain how to install this programming/runtime environment.

1. Cygwin Installation :

As described on its webpage (https://www.cygwin.com), Cygwin consists of GNU and Open Source tools which provide functionality on Windows similar to that of a Linux distribution. Follow the next steps to install Cygwin on Windows 10.

1. Navigate to the installation webpage (https://cygwin.com/install.html) and download the installation file, called *setup-x86_64.exe* (Figure 13).
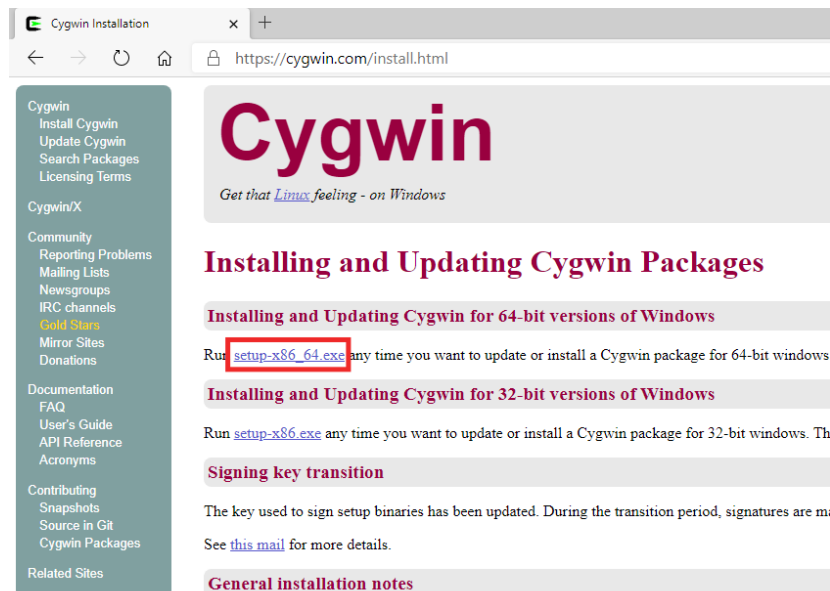
Figure 13. Cygwin installation webpage

2. Execute the setup file in your machine by double-clicking on it (Figure 14). Click **Next** several times, maintaining the default options. The installer will ask you to **Choose a Download Site** (Figure 15), you can choose any one of them.
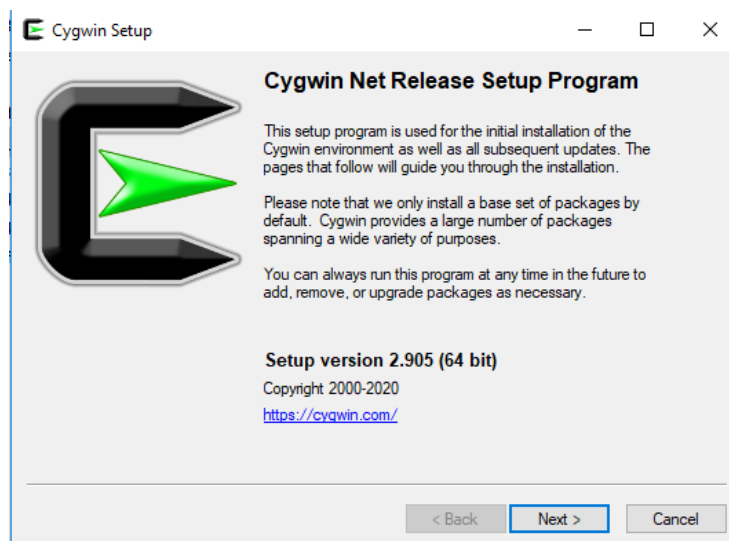
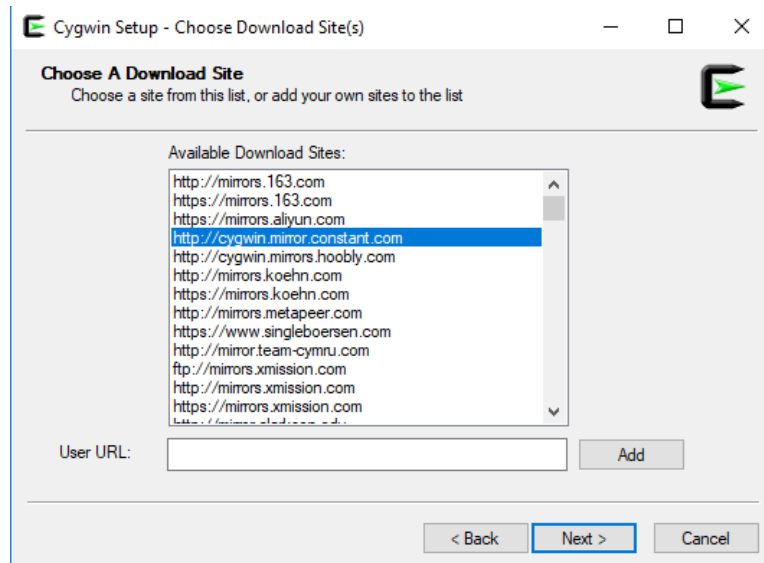**Figure 14. Cygwin installation window**

Figure 15. Choose Download Site

3. After several steps, you will reach the **Select Packages** window (Figure 16). Select the **Full** view, as shown in Figure 16.
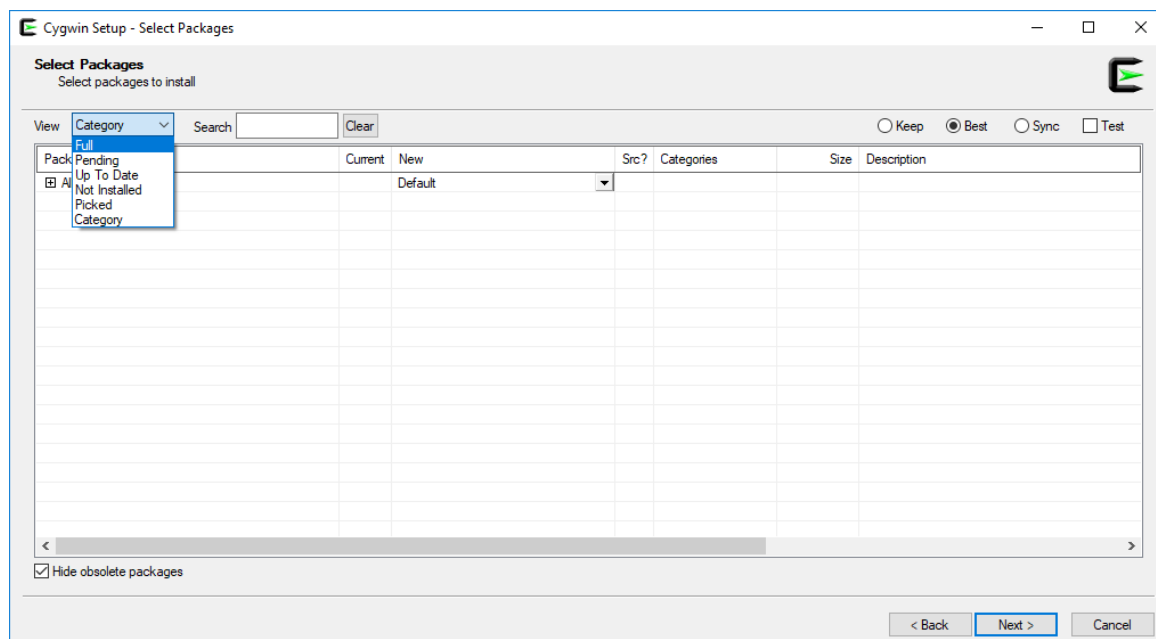

Figure 16. Select Packages window

4. The complete list of packages that you can install will appear (Figure 17). In the **Search** box, select the specific packages that you want to install.
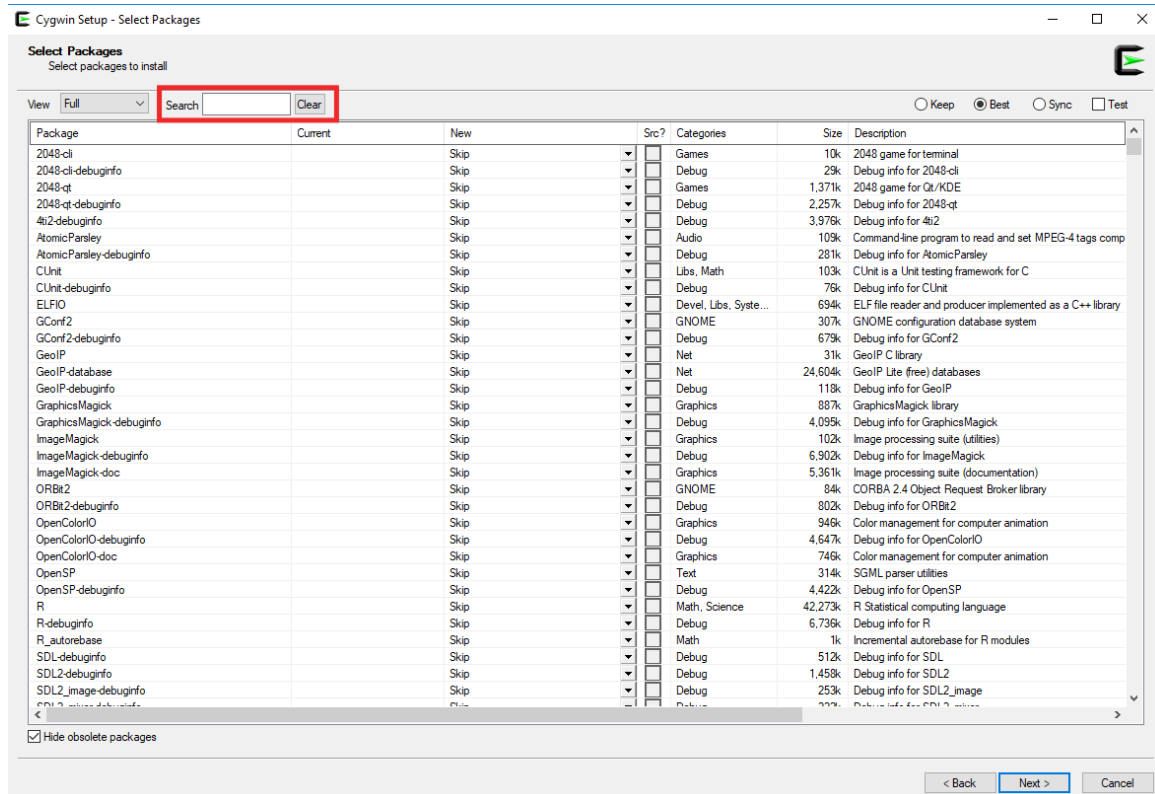
Figure 17. Select Packages window – Full view

To be able to compile Verilator and generate a new simulator binary, you need to install the following packages:

- `git`
- `make`
- `autoconf`
- `gcc-core`
- `gcc-g++`
- `flex`
- `bison`
- `perl`
- `libargp-devel`

Include at least these packages in your Cygwin installation. Select them one-by-one following the steps below (we only show the detailed steps for the first package in the list, `git`; the process is the same for the other packages):

- Look for the `git` package in the **Search** box (Figure 18).
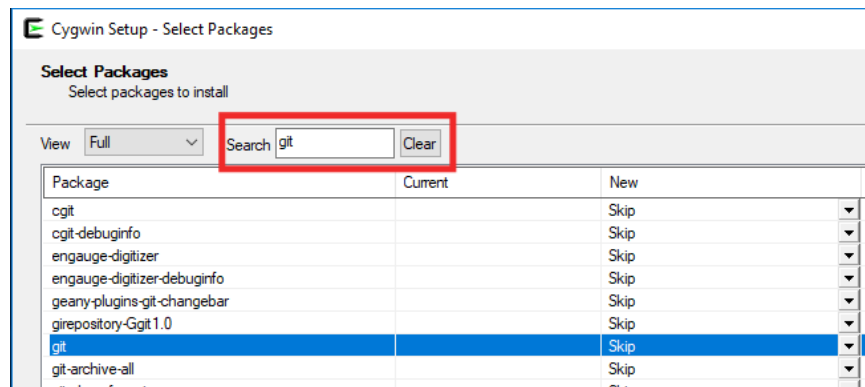
Figure 18. Look for the `git` package

- Select the most up-to-date version in the dropdown menu **and** tick the box (Figure 19).

Figure 19. Select the most up-to-date version and tick the box

- Do the same for the remaining packages in the above list.

But for "gcc-g++" and "gcc-core" packages, install the "10.2.0" version because this version is compatible with the verilator version that we use in these labs.
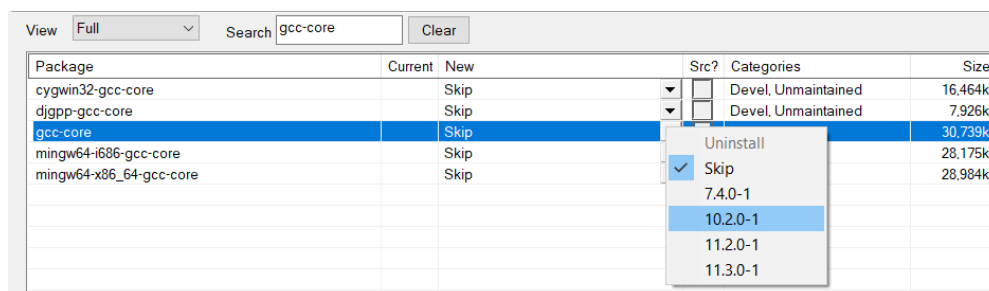

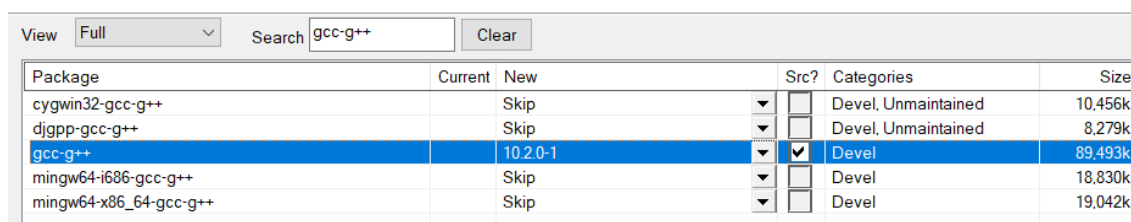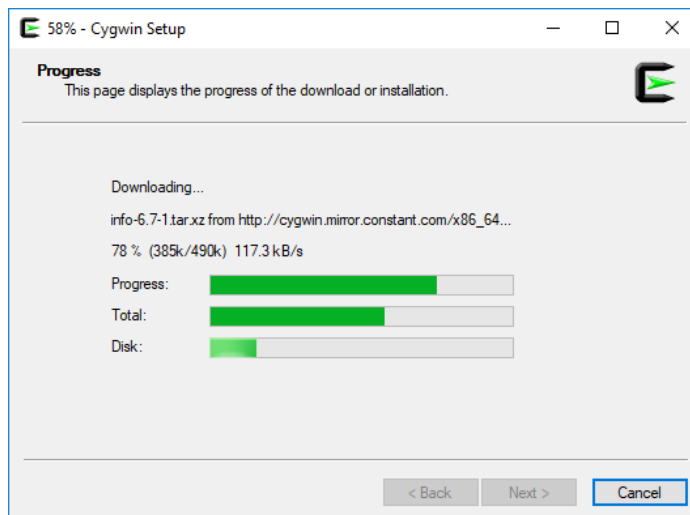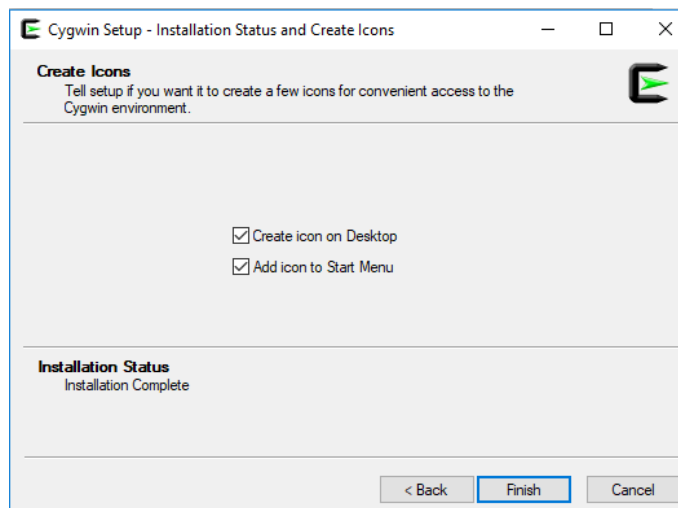**Figure 20. gcc-core version 10.2.0**

Figure 21. gcc-g++ version 10.2.0

5. Once you have selected the nine packages, click **Next** in the subsequent windows to include these packages in your Cygwin installation (the installation process, see Figure 22, may take several minutes) and finalize the installation by clicking Finish (Figure 23).

Figure 22. Cygwin setup

Figure 23. Finish the installation

6. If you need to add a package to your Cygwin installation, repeat steps 2-5 for that package.

2. Verilator Installation in Windows:

Follow the next steps to install Verilator on Windows 10.

1. Open the Cygwin terminal (Figure 24), available on your Windows Desktop or from the Start menu.
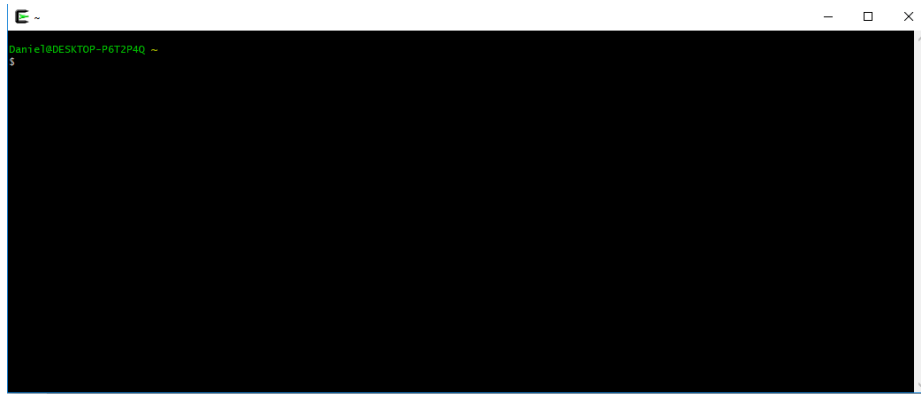
Figure 24. Cygwin terminal

2. Build and install Verilator by following these steps. This may take some time (even hours), depending on the speed of your computer:

```
□  git clone https://github.com/verilator/verilator
□  cd verilator
□  git pull
□  git checkout v4.106
□  autoconf
□  ./configure
□  make
□  make install
```

3. GTKWave Installation in Windows:

GTKWave can be downloaded as a precompiled package from https://sourceforge.net/projects/gtkwave/files/. Look for the most recent Windows package (at the time this document was written, it was called *gtkwave-3.3.100-bin-win64*), and download and unzip (uncompress) it. You can find an executable file called *gtkwave* inside folder *bin*, which you can execute and use in your Windows machine.